Webpage: www.ijaret.org Volume 4, Issue V, May 2016
ISSN 2320-6802

INTERNATIONAL JOURNAL FOR ADVANCE RESEARCH IN ENGINEERING AND TECHNOLOGY

WINGS TO YOUR THOUGHTS…..

An Improved Load Balancing Algorithm for public Cloud Environment

Kiran[1], Talwinder Singh[2]

[1]M.Tech Student, Department of Computer Science, Punjabi University, India
[1]*deep.kiran82@gmail.com*
[3]Assistant Professor, Department of Computer Science, Chandigarh University, India
[3]*talwinder000@gmail.com*

Abstract *Cloud Computing is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction. This Cloud model promotes availability and is composed of five essential characteristics, three service models, and four deployment models. The five essential characteristics are on-demand self service, broad network access, resource pooling, rapid elasticity, and measured service. The three service models are Cloud Software as a Service (SaaS), Cloud Platform as a Service (PaaS), and Cloud Infrastructure as a Service (laaS).The four deployment models are Private Cloud, Community Cloud, Public Cloud, and Hybrid Cloud. In our proposed research work, simulation scenario has been tested using 50 and 150 nodes. The proposed algorithm handle the lease requests (AR, BE and IL) efficiently as compared Anti-starvation and Haizea. The parameters used for comparison are % AR lease, Starvation Threshold and CPU utilization.*

Keywords: *Cloud computing,, Load balancing, Cloud simulator ,priority, Waiting time and Turnaround time.*

1. INTRODUCTION

Cloud computing is a high throughput computing paradigm where large data centres or server farms deliver a large number of computing services to end users. The cloud computing model offers users ubiquitous access to resources through their connected devices. Many clouds are built over the Internet by commercial providers and are accessible by any users who pay for the cloud services through subscriptions. This cloud delivery model is known as public clouds. Many public clouds are available, such as Google App Engine (GAE), Amazon Web Services (AWS) and Microsoft Azure9. The emergence of cloud computing imposes fundamental changes in software and hardware architectures. Cloud architectures, specifically Infrastructure-as-a-Service (IaaS), put more emphasis on the quantity of computing resources or virtual machines (VM) instances. Cloud providers sign Service Level Agreements (SLAs) with end users committing to secure sufficient resources such as CPU, memory and bandwidth for a preset period Efficient VM provisioning depends on the scheduling scheme of the cloud infrastructure that is usually carried by a specialized scheduler that allocates incoming jobs to virtual machines (VMs) examples include Haizea 4, platform ISF 5, VMware vSphere.[1]

2. LEASE MANAGEMENT

Haizea is open-source VM-based lease management architecture. Haizea is a software component that can manage a set of computers (typically a cluster), allowing users to request exclusive use of those resources described in a variety of terms. The fundamental resource provisioning abstraction in Haizea is the lease. Intuitively, a lease is some form of contract where one party agrees to provide a set of resources (an apartment, a car, etc.) to another party. When a user wants to request computational resources from Haizea, it does so in the form of a lease.[7] When applied to computational resources, the lease abstraction is a powerful and general construct with a lot of nuances. Haizea aims to support resource leasing along these three dimensions:[8]

Best-effort lease: Resources are provisioned as soon as they are available.

Advance reservation: AR style leases resources are provisioned during a strictly defined period.

Immediate leases: Resources must be provisioned right now, or not at all.

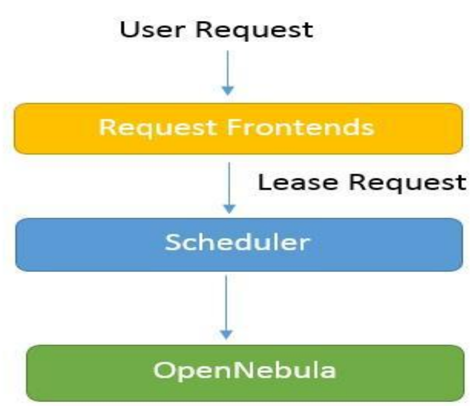

Figure 1: Haizea architecture

3. LITERATURE REVIEW

Mayank Mishra et al. concluded that, the users of cloud services pay only for the amount of resources (a pay-as-use model) used by them. Traditional data centers are provisioned to meet the peak demand, which results in wastage of resources during non-peak periods. To alleviate the above problem, modern-day data centers are shifting to

INTERNATIONAL JOURNAL FOR ADVANCE RESEARCH IN ENGINEERING AND TECHNOLOGY
WINGS TO YOUR THOUGHTS…..

the cloud. The important characteristics of cloud-based data centers are making resources available on demand. The operation and maintenance of the data center lies with the cloud provider. The pay as- use model, that is, users pay only for the services used and hence do not need to be locked into long-term commitments. As a result, a cloud-based solution is an attractive provisioning alternative to exploit the computing- as-service model.[2]

Vijindra et al. presented an algorithm for a cloud computing environment that could automatically allocate resources based on energy optimization methods. Then prove the effectiveness of our algorithm. In the experiments and results analysis, we find that in a practical Cloud Computing Environment, using one whole Cloud node to calculate a single task or job will waste a lot of energy, even when the structure of cloud framework naturally support paralleled process. To deploy an automatic process to find the appropriate CPU frequency, main memory's mode or disk's mode or speed.[3]

Qiang Li and YikeGuo proposed a model for optimization of SLA-based resource schedule in cloud computing based on stochastic integer programming technique. The performance evaluation has been performed by numerical studies and simulation. The experimental result shows that the optimal solution is obtained in a reasonably short time.[4]

Xin Lu, Zilong GU discussed that, by monitoring performance parameters of virtual machines in real time, the overloaded is easily detected once these parameters exceeded the threshold. Quickly finding the nearest idle node by the ant colony algorithm from the resources and starting the virtual machine can bears part of the load and meets these performance and resource requirements of the load. This realizes the load adaptive dynamic resource scheduling in the cloud services platform and achieves the goal of load balancing.[5]

Liang Luo et al. discussed about, a new VM Load Balancing Algorithm is proposed and then implemented in Cloud Computing environment using CloudSim toolkit, in java language. In this algorithm, the VM assigns a varying (different) amount of the available processing power to the individual application services. These VMs of different processing powers, the tasks/requests (application services) are assigned or allocated to the most powerful VM and then to the lowest and so on. we have optimized the given performance parameters such as response time and data processing time, giving an efficient VM Load Balancing algorithm i.e. Weighted Active Load Balancing Algorithm in the Cloud Computing environment[6].

4. METHODOLOGY

The existing anti starvation algorithm prone to starvation problem when the load increases. This leads to the under utilization of resources. The proposed algorithm overcomes this problem by giving a threshold value to the best effort lease. This threshold value is pre defined. By this if the number of rejection of best effort lease is equal to or greater than the threshold value then there is a choice for consumer to convert the best effort lease to advance reservation.

4.1 Installing Haizea

Haizea has been tested only on Unix systems, and the installation instructions are given witha Unix system in mind. However, Haizea includes a small amount of platform-specifc code, and should run fine on other systems with minimal effort. Installing Haizea can be accomplished in four simple steps:

4.1.1 Install dependencies

Haizea has a couple of software dependencies. The main and necessary dependencies are:

- Python 2.5. (http://www.python.org/)
- mxDateTime 3.1.0

4.1.2 Download Haizea

Go to the download page and download the latest version of Haizea. This will be a tarball called haizea-XXX.tar.gz, where XXX will be the version number.

4.1.3 Install Haizea

Go into directory $HAIZEA INST and un-tar the installation package:

- ➢ tarxvzf haizea-XXX.tar.gz

This will create a directory and run the following:

- ➢ Python setup.py install

4.1.4. Verify installation

Haizea includes some sample configuration files and lease request tracefiles that you can use totest Haizea. If you installed Haizea as root, you can run the following to test your installation:

haizea -c /usr/share/haizea/etc/sample_trace.conf

4.2 Proposed Algorithm

In our proposed work we have used negotiation method for improving the overall performance of the algorithm. Extra steps have been removed for assigning leases to the requests.

Algorithm: Proposed Anti Starvation Algorithm

IF lease type== immediate **THEN:**
Allocate the resources to immediate lease
ELSE
Reject immediate lease
IF lease type==BE **THEN:**
Queue BE and set lease state to queue
ELSEIF lease type==AR
Allocate the resources to AR lease
IF suspension value > threshold **THEN**:
Due to the current load on the system, your request would be converted to BE: (C) for Continue, (D) for Decline
IF choice ==C **THEN:**
AR lease converted to BE lease
ELSEIF choices == D **THEN:**
Reject the AR lease
ELSE
Suspend BE lease
Allocate the resources to AR lease

INTERNATIONAL JOURNAL FOR ADVANCE RESEARCH IN ENGINEERING AND TECHNOLOGY
WINGS TO YOUR THOUGHTS…..

5. RESULTS AND EXPLANATION

Simulation scenario consisting of 50 or 150 nodes has been implemented in Haizea using python. Parameters considered for comparison are % AR Lease, starvation threshold and CPU utilization.

Advanced Reservation Lease:- An advance reservation, or AR, lease is a lease that must begin and end at very specific times.

Table 1: % AR Lease

Nodes	IT	%AR Lease (Base)	%AR Lease (Proposed)
50	128	20%	16%
150	512	30%	24%

.

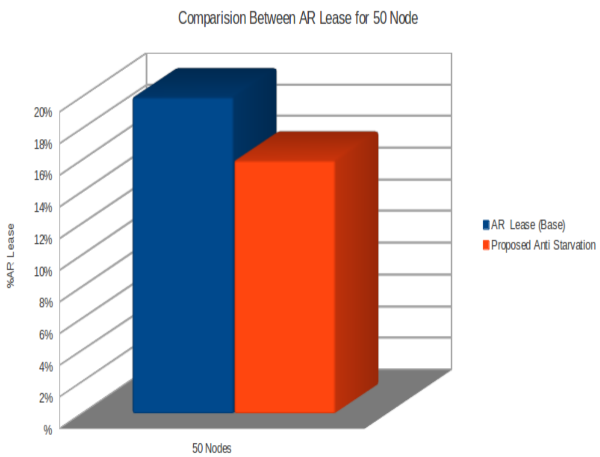

Figure: 1 %AR Lease for 50 Nodes

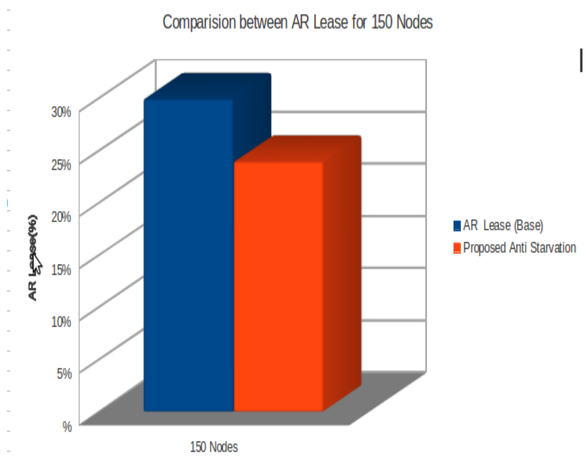

Figure: 2 %AR Lease for 150 Nodes

Starvation threshold: It is the maximum value after which starvation of resources starts occurring. Starvation threshold for the proposed algorithm is 8 which mean it has more resources available.

Table 2: Starvation Threshold

Nodes	IT	Starvation Threshold (Base)	Starvation Threshold (Proposed)
50	128	7	8
150	512	17	21

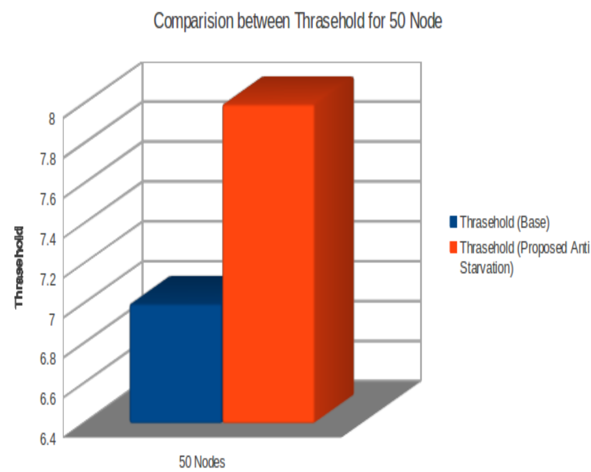

Figure: 3 Starvation thresholds for 50 Nodes

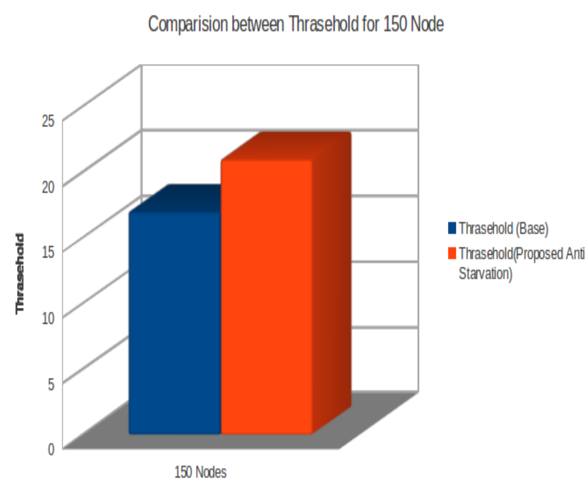

Figure: 4 Starvation thresholds for 150 Nodes

6. CONCLUSION AND FUTURE WORK

In the proposed research work, resource scheduling has been improved in Haizea scheduler to deal with the problem of starvation. The Anti-starvation algorithm has been implemented in Python and tested on a simulated public cloud model built in Haizea. Haizea model provides a set of Immediate, Best-Effort, Dead Line Sensitive, and Advance Reservation scheduling mechanisms. Any of this scheduling mechanism can be used as per the client needs. To deal with

INTERNATIONAL JOURNAL FOR ADVANCE RESEARCH IN ENGINEERING AND TECHNOLOGY
WINGS TO YOUR THOUGHTS…..

problem of efficient resource scheduling in cloud environment we provided a flexible scheduling mechanism based on the load on the server. As the load or the client request on the server will increase our model will switch for equal resource sharing mechanism. The experimental results show that the modified Anti-starvation algorithm outperforms compared to simple anti starvation algorithm. The comparison is shown in the form of graphs consisting of AR lease for 50 nodes and 150 nodes. It also consists of Threshold lease value for 50 nodes and 150 nodes. In future work, more cases can be considered for calculating the performance of anti starvation algorithms, for instance more values for the aging threshold and percentage of AR leases will be considered to get clearer insight into the proposed algorithm performance. We will also try for more number of nodes for various scenarios.

REFERENCES

[1] Hitoshi Matsumoto, Yutaka Ezaki," Dynamic Resource Management in Cloud Environment", July 2011, FUJITSU science & Tech journal, Volume 47, No: 3, page no: 270-276.

[2] Mayank Mishra, Anwesha Das, Purushottam Kulkarni, and Anirudha Sahoo, "Dynamic Resource Management Using Virtual Machine Migrations", Sep 2012, 0163-6804/12, IEEE Communications Magazine, page no: 34-40.

[3] Vijindra and Sudhir Shenai. A, "Survey of Scheduling Issues in Cloud Computing", 2012, ICMOC-2012, 1877-7058, Elsevier Ltd, Doi: 10.1016/j.proeng.2012.06.337, page no: 2881 – 2888.

[4] Qiang Li and Yike Guo, "Optimization of Resource Scheduling in Cloud Computing", 2010, 12th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing, 978-0-7695-4324-6/10, IEEE, DOI 10.1109/SYNASC.2010.8, page no: 315 – 320.

[5] Xin Lu, Zilong Gu, "A Load-Adapative Cloud Resource Scheduling Model Based on Ant Colony Algorithm", 2011, 978-1-61284-204-2/11, Proceedings of IEEE CCIS2011, Page no: 296-300.

[6] Liang Luo, Wenjun Wu, Dichen Di, Fei Zhang, Yizhou Yan, Yaokuan Mao, "A Resource Scheduling Algorithm of Cloud Computing based on Energy Efficient Optimization Methods", 2012, 978-1-4673-2154-9/12, IEEE.

[7] Nathani, S. Chaudhary, and G. Somani., "Policy based resource allocation in IaaS cloud", Future Generation Computer Systems 2011.

[8] Yogita Chawla et al "Dynamically optimised cost based task scheduling in Cloud Computing" Pune university, G. H Raisoni College of Engg & Mgmt. Vol 2, Issue 3, May-June 2013.